

GMX · SECURITY REVIEW

GMX

Fee Automation.

A security review of Fee Automation.

CLIENT	PROJECT	AUDITORS	AUDIT WINDOW	FINAL REPORT
GMX	Fee Automation	Andreas Zottl	August 19 – August 25, 2025	August 25, 2025

§ 00

Contents.

31 pages: overview, confidence ranking, scope, 21 findings, disclaimer, and about Guardian.

§ 01	Overview	P. 03
§ 02	Confidence Ranking	P. 04
§ 03	Scope & Classifications	P. 05
	Files in Scope	P. 06
§ 04	Findings	P. 07
	Main Review · 21 findings	P. 08
§ 05	Disclaimer	P. 30
§ 06	About Guardian	P. 31

§ 01

Overview.

Guardian reviewed Fee Automation across a single review round. The final report includes 1 High, 3 Medium, 5 Low, 12 Informational findings.

AUDIT FIRM	REPOSITORY
Guardian	https://github.com/gmx-io/gmx-automation , https://github.com/gmx-io/gmx-synthetics
CLIENT	METHODOLOGY
GMX	Smart contract · Manual · Main Review
PROJECT	MAIN REVIEW COMMIT
Fee Automation	7a628a30ccdf
PREPARED BY	
Andreas Zottl	
AUDIT WINDOW	
August 19 – August 25, 2025	
FINAL REPORT	
August 25, 2025	

1.1 Confidence Ranking

4 / 5

High Confidence

Guardian assigns a Confidence Ranking of 4 out of 5, combining finding severity, remediation outcome, and the reviewer’s assessment of latent risk. The full scale is on the next page.

1	2	3	4	5
VERY LOW CONFIDENCE	LOW CONFIDENCE	MODERATE CONFIDENCE	HIGH CONFIDENCE	VERY HIGH CONFIDENCE

1.2 Findings at a Glance

SEVERITY	TOTAL	PENDING	ACK	PARTIAL	RESOLVED
■ Critical	0	0	0	0	0
■ High	1	0	0	0	1
■ Medium	3	1	1	0	1
■ Low	5	1	0	0	4
■ Informational	12	3	5	0	4
All severities	21	5	6	0	10

§ 02

Confidence Ranking.

Guardian assigns a project-level confidence score using the scale below. The ranking combines finding severity, remediation outcome, and the reviewer's assessment of latent risk.

CONFIDENCE RANKING	DEFINITION AND RECOMMENDATION	RISK PROFILE
5 Very High Confidence	Codebase is mature, clean, and secure. No High or Critical vulnerabilities were found. Follows modern best practices with high test coverage and thoughtful design. Recommendation: code is highly secure at time of audit.	0 High/Critical findings and few Low/Medium severity findings.
4 High Confidence ASSIGNED	Code is clean, well-structured, and adheres to best practices. Design patterns are sound and test coverage is strong. Recommendation: suitable for deployment after remediations; consider periodic review with changes.	0-1 High/Critical findings per engagement week and little to no Medium severity findings.
3 Moderate Confidence	Code is functional but shows gaps in defensive programming or test coverage. Recommendation: remediate the findings and re-review the changes before deployment.	Some High/Critical findings, with various Medium severity findings.
2 Low Confidence	Recurring issues suggest systemic gaps rather than isolated mistakes. Recommendation: remediate, then commission a follow-up review at a frozen commit.	Over 6 High and Critical findings across the engagement.
1 Very Low Confidence	The volume and severity of findings indicate the codebase is not ready for production. Recommendation: substantial rework, then a full re-review.	Over 12 High and Critical findings across the engagement.

§ 03

Scope & Classifications.

Guardian reviewed Fee Automation in <https://github.com/gmx-io/gmx-automation>, <https://github.com/gmx-io/gmx-synthetics>. The scope includes 137 source lines across 1 file.

3.1 Severity Matrix

	IMP · HIGH	IMP · MED	IMP · LOW
LIK · HIGH	Critical	High	Medium
LIK · MED	High	Medium	Low
LIK · LOW	Medium	Low	Low

IMPACT · HIGH

Significant loss of assets, harm to a group of users, or core protocol functionality is disrupted.

IMPACT · MEDIUM

Small amount of funds can be lost, or ancillary functionality is affected. Users may experience reduced or delayed receipt.

IMPACT · LOW

Unexpected behavior in some functionality that is notable but does not meet the criteria for higher severity.

LIKELIHOOD · HIGH

Possible with reasonable assumptions that mimic on-chain conditions; cost is low compared to amount gained.

LIKELIHOOD · MEDIUM

Possible only in uncommon cases or requires a large amount of capital relative to disruption caused.

LIKELIHOOD · LOW

Unlikely to ever occur in production.

Files in Scope.

FILE	NSLOC	TOTAL
contracts/contributor/ContributorHandler.sol	137	182

§ 04

Findings.

Grouped by round. Visible IDs reset by severity within each round, so a finding is named the same way in the report as it was in the channel it was written in. Status is where each finding ended up; its own page says how.

ID	TITLE	CATEGORY	SEVERITY	STATUS	PAGE
MAIN REVIEW					
H-01	Weekly Settlement Window Misalignment	Rewards	HIGH	RESOLVED	p. 8
M-01	Keepers Can Steal Fees	Logical Error	MEDIUM	PENDING	p. 9
M-02	Zero Block Confirmations For Event Triggers	Configuration	MEDIUM	RESOLVED	p. 10
M-03	Bridged GMX Trigger Can Be Abused	DoS	MEDIUM	ACKNOWLEDGED	p. 11
L-01	FeeDistributor Can Permanently Stuck	Logical Error	LOW	PENDING	p. 12
L-02	Missing Token Validation	Validation	LOW	RESOLVED	p. 13
L-03	Missing Action In Timelock Config	Logical Error	LOW	RESOLVED	p. 14
L-04	Unclear Log: "No relevant event"	Best Practices	LOW	RESOLVED	p. 15
L-05	Cron Expression Compatibility	Best Practices	LOW	RESOLVED	p. 16
I-01	Redundant Check In initiateDistribute	Superfluous Code	INFORMATIONAL	PENDING	p. 17
I-02	Invalid NatSpec	Superfluous Code	INFORMATIONAL	RESOLVED	p. 18
I-03	Treasury Funds Risk Exposure	Warning	INFORMATIONAL	RESOLVED	p. 19
I-04	Blacklisted Token Transfers	Warning	INFORMATIONAL	ACKNOWLEDGED	p. 20
I-05	Static Absolute Contributor Amounts	Configuration	INFORMATIONAL	PENDING	p. 21
I-06	Missing LayerZero Read Channels	Configuration	INFORMATIONAL	ACKNOWLEDGED	p. 22
I-07	On-Chain Test Runs Required	Best Practices	INFORMATIONAL	ACKNOWLEDGED	p. 23
I-08	Fee Distribution Period Overlap	Warning	INFORMATIONAL	RESOLVED	p. 24
I-09	Fee Distribution Requires Synchronous Execution	Warning	INFORMATIONAL	RESOLVED	p. 26
I-10	Bridging Excess GMX Requires Native Fee	Warning	INFORMATIONAL	PENDING	p. 27
I-11	Cross-chain Overrides/Read	Configuration	INFORMATIONAL	ACKNOWLEDGED	p. 28
I-12	Reliance On Gelato Key-Store Values	Warning	INFORMATIONAL	ACKNOWLEDGED	p. 29

HIGH **H-01**

RESOLVED

Weekly Settlement Window Misalignment

CATEGORY Rewards

LOCATION <https://github.com/gmx-io/gmx-automation/blob/7a628a30ccdf708c1630994fba24ade15da4ec53/src/web3-functions/feeDistribution/feeDistribution.ts#L104-L105>

REVIEW Main Review

DESCRIPTION

The Web3Function ends the accounting window at the **trigger block timestamp**, not at a fixed weekly boundary.

- When `eventName === "FeeDistributionDataReceived"` and `isBridgingCompleted === true`, the script computes rewards up to the LZ receive/process time, not the intended week end.
- When `eventName === "FeeDistributionBridgedGmxReceived"`, it cuts the window at the GMX-bridge arrival time.

This drifts from the Wednesday→Wednesday cadence and produces partial/incorrect weekly rewards.

RECOMMENDATION

Consider using canonical `{from, to}` **weekly window** (e.g., Wed 00:00 UTC → next Wed 00:00 UTC) like its done in `processV1` fees.

<https://github.com/gmx-io/gmx-automation/blob/7a628a30ccdf708c1630994fba24ade15da4ec53/src/domain/fee/feeDistributionService.ts#L130-L131>

RESOLUTION

Status: Resolved.

<https://github.com/gmx-io/gmx-automation/blob/audit-finding-updates/src/domain/fee/feeDistributionService.ts#L160>

MEDIUM **M-01**

PENDING

Keepers Can Steal Fees

CATEGORY	Logical Error
LOCATION	https://github.com/gmx-io/gmx-synthetics/blob/8d920064b47c83b020ccf301b7df507832380b57/contracts/fee/FeeDistributorUtils.sol#L36-L37
REVIEW	Main Review

— DESCRIPTION

Since the keeper's raw ETH balance is used for comparison against the target, they can move funds to claim more than what is actually attributed to them.

Additionally, because keepers are responsible for performing certain on-chain actions, they always need to maintain some funds to run those operations.

The calculation target – current balance fails to account for the operating balance required for those actions.

— RECOMMENDATION

Consider sending the amounts directly in `initiateDistribution` instead of relying on the keeper's balance, or be aware that in this case, keepers could theoretically always claim up to

`FEE_DISTRIBUTOR_KEEPER_COSTS`

— RESOLUTION

Status: Pending.

MEDIUM **M-02**

RESOLVED

Zero Block Confirmations For Event Triggers

CATEGORY Configuration

LOCATION https://github.com/gmx-io/gmx-automation/blob/7a628a30ccdf708c1630994fba24ade15da4ec53/scripts/createTask_feeDistribution.ts#L69-L70 https://github.com/gmx-io/gmx-automation/blob/7a628a30ccdf708c1630994fba24ade15da4ec53/scripts/createTask_feeDistributor_bridgedGmxReceived.ts#L32-L33

REVIEW Main Review

DESCRIPTION

GMX automation scripts configure Gelato Web3 Function triggers with `blockConfirmations = 0`. If a chain reorg occurs, events may be replayed or missed, leading to duplicated distributions or inconsistent state.

RECOMMENDATION

Set a chain-appropriate number of block confirmations for Gelato triggers to reduce reorg-induced duplication.

Note that if block confirmations are used, it would trigger web3 function with small delay, hence it could create inconsistency if exact timestamp is used in calculations.

RESOLUTION

Status: Resolved.

<https://github.com/gmx-io/gmx-automation/blob/audit-finding-updates/src/lib/number.ts#L28>

MEDIUM **M-03**

ACKNOWLEDGED

Bridged GMX Trigger Can Be Abused

CATEGORY DoS

LOCATION https://github.com/gmx-io/gmx-automation/blob/7a628a30ccdf708c1630994fba24ade15da4ec53/scripts/createTask_feeDistributor_bridgedGmxReceived.ts#L29-L30

REVIEW Main Review

— DESCRIPTION

The `FeeDistributorBridgedGmxReceived` automation is triggered by *any* GMX transfer into the fee distributor vault. Since there is no minimum amount check, an attacker could grief the keeper by sending 1 wei of GMX, repeatedly draining Gelato keeper balances through redundant executions.

— RECOMMENDATION

Add a threshold check in event trigger (e.g., require at least X GMX received) before allowing the trigger to fire. This ensures malicious micro-transfers only result in wasted donations rather than draining keeper resources.

— RESOLUTION

Status: Acknowledged.

LOW **L-01**

PENDING

FeeDistributor Can Permanently Stuck

CATEGORY	Logical Error
LOCATION	Global
REVIEW	Main Review

— DESCRIPTION

The `FeeDistributor` contract uses `validateReadResponseTimestamp` to enforce that LayerZero (LZ) read responses are received within the configured maximum delay (`FEE_DISTRIBUTOR_MAX_READ_RESPONSE_DELAY`). Similarly, Gelato keeper calls are expected within this time frame.

If a response or keeper call arrives late, the corresponding method (e.g. `processLzReceive`) will revert, leaving the contract stuck in the `Initiated` state. In this state:

- `initiateDistribute` cannot be called again (requires state `None`).
- No other function transitions state back to `None`.
- As a result, the distribution process cannot be restarted.

This creates a **deadlock scenario** where a single delayed or missing read/keeper execution can halt fee distributions indefinitely.

Weekly fee distribution may become permanently blocked if one LZ read or Gelato execution is delayed. Recovery is impossible without a contract upgrade or manual intervention.

— RECOMMENDATION

Consider allowing a reset of the state back to `None` if the system has been in `Initiated` (or `ReadDataReceived`) longer than `FEE_DISTRIBUTOR_MAX_READ_RESPONSE_DELAY`.

This could be implemented as a restricted function callable by the fee distribution keeper or governance.

— RESOLUTION

Status: Pending.

LOW **L-02**

RESOLVED

Missing Token Validation

CATEGORY Validation

LOCATION <https://github.com/gmx-io/gmx-synthetics/blob/8d920064b47c83b020ccf301b7df507832380b57/contracts/contributor/ContributorHandler.sol#L76-L77>

REVIEW Main Review

DESCRIPTION

The function `setMaxTotalContributorTokenAmount` does not verify whether the provided token exists in the `DataStore`. As a result, an arbitrary token can be passed and a max value set, effectively bypassing the intended max validation logic. `_validateMaxContributorTokenAmounts`

This contrasts with `setContributorAmount`, where the token existence check is correctly enforced — suggesting the validation was unintentionally omitted in `setMaxTotalContributorTokenAmount`.

RECOMMENDATION

Add a token existence check in `setMaxTotalContributorTokenAmount`.

RESOLUTION

Status: Resolved.

<https://github.com/gmx-io/gmx-synthetics/blob/fee-automation/contracts/contributor/ContributorHandler.sol#L87> (<http://github.com/gmx-io/gmx-synthetics/blob/fee-automation/contracts/contributor/ContributorHandler.sol#L87>)

LOW **L-03**

RESOLVED

Missing Action In Timelock Config

CATEGORY	Logical Error
LOCATION	https://github.com/gmx-io/gmx-synthetics/blob/8d920064b47c83b020ccf301b7df507832380b57/contracts/contributor/ContributorHandler.sol#L76-L77
REVIEW	Main Review

— DESCRIPTION

The function `setMaxTotalContributorTokenAmount` can only be called by the controller, but it does not have a corresponding `set` action in the `**TimelockConfig**`. Hence, this parameter is `**not configurable.**`

A similar concern applies to:

- `setMinContributorPaymentInterval`

— RECOMMENDATION

Consider adding corresponding timelock functions

— RESOLUTION

Status: Resolved.

<https://github.com/gmx-io/gmx-synthetics/blob/fee-automation/contracts/config/TimelockConfig.sol#L477>

LOW **L-04**

RESOLVED

Unclear Log: "No relevant event"

CATEGORY Best Practices

LOCATION <https://github.com/gmx-io/gmx-automation/blob/7a628a30ccdf708c1630994fba24ade15da4ec53/src/web3-functions/feeDistribution/feeDistribution.ts#L187-L188>

REVIEW Main Review

DESCRIPTION

Current branch returns:

```
``typescript
{ canExec: false, message: No relevant event found: ${eventName} }
`
even when FeeDistributionDataReceived is the correct event but isBridgingCompleted === false`.
This masks the true state (i.e., waiting for bridge completion).
```

RECOMMENDATION

Add a dedicated branch and log:

```
```typescript
{ canExec: false, message: "FeeDistributionDataReceived seen; bridging not completed — waiting" }
...

```

Reserve "No relevant event" strictly for **unknown topics**.

### RESOLUTION

Status: Resolved.

<https://github.com/gmx-io/gmx-automation/blob/audit-finding-updates/src/web3-functions/feeDistribution/feeDistribution.ts#L148>

LOW **L-05**

RESOLVED

# Cron Expression Compatibility

CATEGORY Best Practices

LOCATION [https://github.com/gmx-io/gmx-automation/blob/7a628a30ccdf708c1630994fba24ade15da4ec53/scripts/createTask\\_feeDistributor\\_initiateDistribute.ts#L20-L21](https://github.com/gmx-io/gmx-automation/blob/7a628a30ccdf708c1630994fba24ade15da4ec53/scripts/createTask_feeDistributor_initiateDistribute.ts#L20-L21)

REVIEW Main Review

## DESCRIPTION

The init automation trigger uses:

```
typescript
trigger: {
 type: TriggerType.CRON,
 cron: "0 0 * * WED", // @audit check
}
```

While "WED" is valid in some cron implementations, Gelato's docs only show numeric syntax (0 0 \* \* 3 for Wednesday). If "WED" is unsupported, the function may never trigger.

## RECOMMENDATION

- Confirm with Gelato whether day-of-week abbreviations are supported.
- Run a test execution to ensure `initiateDistribute()` fires as expected.
- If uncertain, switch to numeric syntax (0 0 \* \* 3).
- Verify whether Gelato treats Monday as 0 or 1 in numeric cron.

## RESOLUTION

Status: Resolved.

[https://github.com/gmx-io/gmx-automation/blob/audit-finding-updates/scripts/createTask\\_feeDistributor\\_initiateDistribute.ts#L20](https://github.com/gmx-io/gmx-automation/blob/audit-finding-updates/scripts/createTask_feeDistributor_initiateDistribute.ts#L20)

## INFORMATIONAL I-01

PENDING

# Redundant Check In `initiateDistribute`

CATEGORY	Superfluous Code
LOCATION	<a href="https://github.com/gmx-io/gmx-synthetics/blob/8d920064b47c83b020ccf301b7df507832380b57/contracts/fee/FeeDistributor.sol#L75-L79">https://github.com/gmx-io/gmx-synthetics/blob/8d920064b47c83b020ccf301b7df507832380b57/contracts/fee/FeeDistributor.sol#L75-L79</a>
REVIEW	Main Review

## DESCRIPTION

The `initiateDistribute` function checks that `getAddress(Keys.FEE_RECEIVER)` equals `address(feeDistributorVault)` and reverts otherwise.

Checking a configured variable against another configured variable is a redundant operation.

## RECOMMENDATION

Consider saving the address only once and always use the same.

## RESOLUTION

Status: Pending.

INFORMATIONAL I-02

RESOLVED

# Invalid NatSpec

CATEGORY	Superfluous Code
LOCATION	<a href="https://github.com/gmx-io/gmx-synthetics/blob/8d920064b47c83b020ccf301b7df507832380b57/contracts/fee/FeeHandler.sol#L52">https://github.com/gmx-io/gmx-synthetics/blob/8d920064b47c83b020ccf301b7df507832380b57/contracts/fee/FeeHandler.sol#L52</a>
REVIEW	Main Review

## DESCRIPTION

The NatSpec of the `withdrawFees` function talks about a `marketTokens` parameter that does not exist.

## RECOMMENDATION

Consider removing that comment.

## RESOLUTION

Status: Resolved.

<https://github.com/gmx-io/gmx-synthetics/blob/fee-automation/contracts/fee/FeeHandler.sol#L52>

## INFORMATIONAL I-03

RESOLVED

# Treasury Funds Risk Exposure

CATEGORY	Warning
LOCATION	Global
REVIEW	Main Review

## DESCRIPTION

In discussion, the GMX team indicated that the `vault` address in `ContributorHandler` could potentially be the DAO treasury, a multisig, or one of the v1 timelocks.

If the vault is set to the **DAO treasury** (or any address holding more funds than those intended for contributor distribution), all of those funds would effectively be **exposed** to the contributor keeper, controller, and contributor distributor.

Since keepers are expected to be managed by automated scripts, this **greatly increases the risk surface**. A similar pattern exists in the FeeDistributor:

- In `finalizeWntForTreasury` ([L557-L558])(<https://github.com/gmx-io/gmx-synthetics/blob/8d920064b47c83b020ccf301b7df507832380b57/contracts/fee/FeeDistributor.sol#L557-L558>), the GMX treasury would need to approve balances to the FeeDistributor.

Thus, both components share the same underlying problem: core treasury funds are unnecessarily exposed through distribution mechanisms.

## RECOMMENDATION

- Explicitly restrict the vault to a **dedicated distribution contract** or **dedicated multisig**, which should only hold the funds required for periodic distributions.
- Avoid using the DAO treasury (or any address with significant unrelated balances) directly in contributor or fee distribution

## RESOLUTION

Status: Resolved.

<https://github.com/gmx-io/gmx-synthetics/blob/fee-automation/contracts/data/Keys.sol#L474>

## INFORMATIONAL I-04

ACKNOWLEDGED

# Blacklisted Token Transfers

CATEGORY	Warning
LOCATION	<a href="https://github.com/gmx-io/gmx-synthetics/blob/8d920064b47c83b020ccf301b7df507832380b57/contracts/contributor/ContributorHandler.sol#L113-L114">https://github.com/gmx-io/gmx-synthetics/blob/8d920064b47c83b020ccf301b7df507832380b57/contracts/contributor/ContributorHandler.sol#L113-L114</a>
REVIEW	Main Review

## DESCRIPTION

Some ERC20 tokens revert on zero-balance or blacklisted transfers, meaning a single problematic contributor could cause all payments for all tokens and contributors to fail, though this risk is mitigated as contributor keeper can remove such contributors.

## RECOMMENDATION

Beware of this scenario.

## RESOLUTION

Status: Acknowledged.

INFORMATIONAL I-05

PENDING

# Static Absolute Contributor Amounts

CATEGORY	Configuration
LOCATION	ContributorHandler.sol
REVIEW	Main Review

## DESCRIPTION

In ContributorHandler, contributor payments are currently set using absolute token amounts. This design is rigid and introduces problems under varying vault balances:

If the vault balance is less than the total contributor amounts, sendPayments would revert.

If the vault balance is greater than the total contributor amounts, the surplus remains unallocated.

Since setContributorAmount applies immediately (pre-action) and persists post-action, it would be applicable for time spent from last payment to present.

## RECOMMENDATION

**\*\*Execute the flow below for the current code:\*\***

1. Calculate contributor amounts based on what you intend to distribute in the upcoming payment.
2. Call `setContributorAmount` for all contributors (multicall is fine).
3. Call `sendPayments`.
4. \*(Optional but recommended)\* Reset all contributor amounts to zero afterward.

## RESOLUTION

Status: Pending.

INFORMATIONAL I-06

ACKNOWLEDGED

# Missing LayerZero Read Channels

CATEGORY Configuration  
LOCATION Global  
REVIEW Main Review

## DESCRIPTION

As per our communication with GMX, they have future plans for using Botonix alongside other deployments for fee calculation and distribution. However, review of the LayerZero deployments shows that Botonix does not have read channels enabled. In addition, Botonix is not listed as a target data chain for either AVAX or ARB under any DVN provider. As a result, cross-chain read communication between these networks is currently not supported:

- ARB/AVAX cannot read data from Botonix.
- Botonix cannot read data from ARB/AVAX.

This limitation means that critical reads, such as fetching total staked or supply values, cannot be performed under current LayerZero read channel support.

## RECOMMENDATION

Beware that Botonix state can't be used for this automatic fee distribution until read channels are supported there.

## RESOLUTION

Status: Acknowledged.

INFORMATIONAL I-07

ACKNOWLEDGED

# On-Chain Test Runs Required

CATEGORY	Best Practices
LOCATION	Global
REVIEW	Main Review

## — DESCRIPTION

Given the involvement of third-party integrations with opaque code, such as **LayerZero DVNs**, it is critical that the flow is first tested on **testnet** (or on mainnet with trial configs and restricted access). Certain configurations can only be reliably validated through **on-chain traces**, making this step essential.

## — RECOMMENDATION

- Conduct a trial run of fee distribution on testnet/mainnet for both AVAX and ARB.
- Share the resulting on-chain traces with us for review and verification of correctness.
- Pay particular attention to gas limits for LayerZero actions, as these are best validated directly on-chain.

### Notes For Guardian Team During Remediations

Check returnDataSize behaviour onchain, +8 requirement

Check bridgeGMX flow, when layerzeroOFT = gmx

## — RESOLUTION

Status: Acknowledged.

## INFORMATIONAL I-08

RESOLVED

# Fee Distribution Period Overlap

CATEGORY Warning

LOCATION gmx-synthetics/contracts/fee/FeeDistributor.sol at 8d920064b47c83b020ccf301b7df507832380b57 · gmx-io/gmx-synthetics (https://github.com/gmx-io/gmx-synthetics/blob/8d920064b47c83b020ccf301b7df507832380b57/contracts/fee/FeeDistributor.sol#L267-L268)

REVIEW Main Review

## DESCRIPTION

The current check in `FeeDistributor.sol` only verifies that the calculated amount meets the minimum bridging requirement:

```
```solidity
// if the calculated amount doesn't meet the min bridging requirement, revert
if (feeAmountGmxCurrentChain < minRequiredGmxAmount) {
    revert Errors.BridgedAmountNotSufficient(minRequiredGmxAmount, feeAmountGmxCurrentChain);
}
```

However, this check does not account for scenarios involving overlapping fee distribution periods:

- Suppose there are two periods: `**x → y**` and `**y → z**`.
- Distribution for `**x → y**` is initiated but insufficient funds are available on-chain.
- While waiting for bridged GMX to arrive, the system progresses into `**y → z**`, during which new fees may already have been deposited into the vault.

This can lead to situations where:

```
`plain text
vault balance > fundsOnChain + fundsBridged
``
```

because the vault balance now includes funds from a `**future period**`.

While not a critical vulnerability, it could cause `**confusing or inconsistent assertions**` when reviewing balances and tracing distributions.

RECOMMENDATION

- Be aware of this behavior when designing assertions and flows.
- Alternatively, document this limitation clearly, as it may not be a significant issue but could affect operational clarity.

I-08 Fee Distribution Period Overlap

CONTINUED

— RESOLUTION

Status: Resolved.

<https://github.com/gmx-io/gmx-synthetics/blob/fee-automation/contracts/fee/FeeDistributor.sol#L75>

INFORMATIONAL I-09

RESOLVED

Fee Distribution Requires Synchronous Execution

CATEGORY	Warning
LOCATION	Global
REVIEW	Main Review

DESCRIPTION

The current fee distribution design assumes that all steps are executed **synchronously across all chains** for a given state.

This reliance on synchronous execution across various chains introduces operational risk if one chain or keeper on one chain experiences downtime

RECOMMENDATION

- Always execute fee distribution actions **in sync across all chains**.
- Establish a **contingency plan** for scenarios where one chain (e.g., Arbitrum) experiences downtime or delayed execution.
- Share this contingency plan with us for review to ensure consistency and operational safety.

RESOLUTION

Status: Resolved.

<https://github.com/gmx-io/gmx-synthetics/blob/fee-automation/contracts/fee/FeeDistributor.sol#L75>

INFORMATIONAL I-10

PENDING

Bridging Excess GMX Requires Native Fee

CATEGORY Warning

LOCATION gmx-synthetics/contracts/fee/FeeDistributor.sol at 8d920064b47c83b020ccf301b7df507832380b57 · gmx-io/gmx-synthetics (https://github.com/gmx-io/gmx-synthetics/blob/8d920064b47c83b020ccf301b7df507832380b57/contracts/fee/FeeDistributor.sol#L479-L480)

REVIEW Main Review

DESCRIPTION

In `FeeDistributor.sol` ([L479-L480])(https://github.com/gmx-io/gmx-synthetics/blob/8d920064b47c83b020ccf301b7df507832380b57/contracts/fee/FeeDistributor.sol#L479-L480)), bridging excess GMX requires a **native fee** to be paid by the contract itself. This means the contract must always hold some **native ETH balance** (or the relevant native token on other chains) in order to successfully perform cross-chain sends.

RECOMMENDATION

Ensure that the contract maintains a **minimum buffer of native ETH** to cover bridging fees.

RESOLUTION

Status: Pending.

INFORMATIONAL I-11

ACKNOWLEDGED

Cross-chain Overrides/Read

CATEGORY	Configuration
LOCATION	https://github.com/gmx-io/gmx-automation/blob/7a628a30ccdf708c1630994fba24ade15da4ec53/src/domain/fee/feeDistributionService.ts#L673-L674
REVIEW	Main Review

— DESCRIPTION

Gelato's KV storage is per task, not auto-namespaced by chain. If the same task is configured for AVAX and ARB, keys like `distributionData`, `wntPrice`, `gmxPrice`, `fromTimestamp` can be overwritten by another chain's run.

This could lead to one's chains data used by other, inside `FeeDistributionCompleted` event processing

— RECOMMENDATION

Consider configuring one Gelato task per chain.

Also consider validating the the stored `chainId` before use/

If you must share a task, namespace all keys by chain id.

— RESOLUTION

Status: Acknowledged.

INFORMATIONAL I-12

ACKNOWLEDGED

Reliance On Gelato Key-Store Values

CATEGORY Warning
LOCATION Global
REVIEW Main Review

— DESCRIPTION

GMX automation relies on key-store values held by Gelato for Web3 Function execution. If an attacker gains access to the Gelato secret store (via leaked secrets, web2 compromise, or Gelato breach), they could claim all fees for themselves.

— RECOMMENDATION

Beware of this risk or consider adding safeguards in script and in contracts for all transfers.
(That at max only certain amount could be transferred for each payment, reducing exposure)
Same risk applies for Contributor Payments.
Or in general fund source of funds, with only whats required for 2 weeks.
So that transfer out is capped.

— RESOLUTION

Status: Acknowledged.

§ 05

Disclaimer.

This report is not, nor should be considered, an "endorsement" or "disapproval" of any particular project or team. This report is not, nor should be considered, an indication of the economics or value of any "product" or "asset" created by any team or project that contracts Guardian to perform a security assessment. This report does not provide any warranty or guarantee regarding the absolute bug-free nature of the technology analyzed, nor do they provide any indication of the technologies proprietors, business, business model or legal compliance.

This report should not be used in any way to make decisions around investment or involvement with any particular project. This report in no way provides investment advice, nor should be leveraged as investment advice of any sort. This report represents an extensive assessing process intending to help our customers increase the quality of their code while reducing the high level of risk presented by cryptographic tokens and blockchain technology.

Blockchain technology and cryptographic assets present a high level of ongoing risk. Guardian's position is that each company and individual are responsible for their own due diligence and continuous security. Guardian's goal is to help reduce the attack vectors and the high level of variance associated with utilizing new and consistently changing technologies, and in no way claims any guarantee of security or functionality of the technology we agree to analyze.

The assessment services provided by Guardian is subject to dependencies and under continuing development. You agree that your access and/or use, including but not limited to any services, reports, and materials, will be at your sole risk on an as-is, where-is, and as-available basis.

Cryptographic tokens are emergent technologies and carry with them high levels of technical risk and uncertainty. The assessment reports could include false positives, false negatives, and other unpredictable results. The services may access, and depend upon, multiple layers of third-parties.

Notice that smart contracts deployed on the blockchain are not resistant from internal/external exploit. Notice that active smart contract owner privileges constitute an elevated impact to any smart contract's safety and security. Therefore, Guardian does not guarantee the explicit security of the audited smart contract, regardless of the verdict.

§ 06

About Guardian.

Guardian is a smart contract security firm. Every engagement is worked by two independent teams competing on the same scope, each with its own tooling and methods, so no single reviewer's blind spot survives.

\$45B+DIGITAL ASSETS UNDER
REVIEW**200+**

ENGAGEMENTS COMPLETED

500+CRITICAL & HIGH FINDINGS
REPORTED

Reach us at audits@guardianaudits.com.